(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号
特開2000－224155
（P2000－224155A）
(43)公開日　平成12年８月11日(2000.8.11)

(51)Int.Cl.⁷	識別記号	ＦＩ		テーマコード（参考）
Ｈ０４Ｌ　9/08		Ｈ０４Ｌ　9/00	６０１Ｂ	５Ｊ１０４
9/16			６０１Ｅ	５Ｋ０３０
12/18			６４３	９Ａ００１
		11/18		

審査請求　未請求　請求項の数４　ＯＬ　（全 10 頁）

(21)出願番号　特願平11－20190

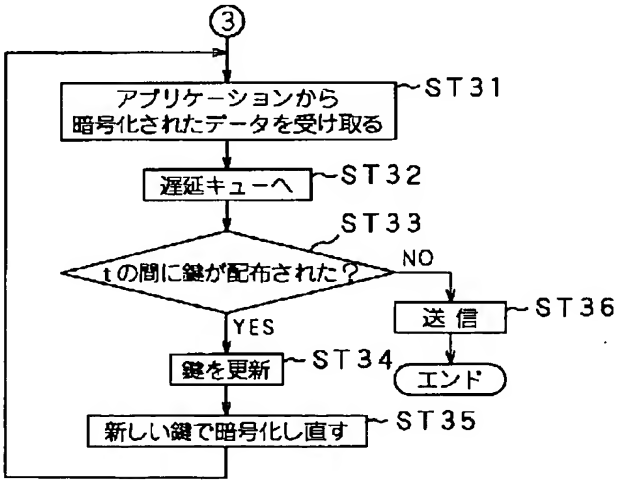
(22)出願日　平成11年１月28日(1999.1.28)

(71)出願人　000002185
ソニー株式会社
東京都品川区北品川６丁目７番35号
(72)発明者　塩野崎　敦
東京都品川区東五反田３丁目14番13号　ソニーコンピュータサイエンス研究所内
(72)発明者　濱野　淳史
神奈川県横浜市港北区日吉３－14－１　慶應義塾大学大学院内
(74)代理人　100067736
弁理士　小池　晃　（外２名）

最終頁に続く

(54)【発明の名称】　ネットワークシステム及びデータ送受信方法

(57)【要約】
【課題】　鍵の配布や鍵を用いたデータの送受信の順番に関係なく、リアルタイム性を維持してマルチキャスト通信を行う。
【解決手段】　遅延キューにデータが供給されてから出力するまでの時間ｔの間に共有鍵が配布されたかを判定する（ステップＳＴ３３）。共有鍵が配布されたと判定したときは共有鍵を更新し（ステップＳＴ３４）、データの暗号化をし直す（ステップＳＴ３５）。また、共有鍵が配布されていないと判定したときはそのデータをそのまま送信する（ステップＳＴ３６）。



【特許請求の範囲】

【請求項１】　鍵サーバと複数のクライアントからなり、上記鍵サーバはクライアントが変更される毎に新たな共有鍵を変更後の各クライアントに送信し、各クライアントは配布された共有鍵を用いてデータを暗号化して暗号化済みのデータに対して鍵サーバと各クライアント間の因果順序を示す因果情報を付加して送受信を行うことで、共有鍵と暗号化済みのデータとを対応付けてデータの復号を行うネットワークシステムにおいて、

上記各クライアントは、

上記鍵サーバに問い合わせをして当該鍵サーバから上記各クライアントに上記共有鍵が送信されるまでの保証されている遅延時間を算出する算出手段と、

送信対象となる暗号化済みのデータに対して、上記算出手段により算出された遅延時間に相当する時間の遅延処理を施す遅延手段と、

上記遅延手段の遅延処理中に上記鍵サーバから新たな鍵が配布されたときは、上記新たな鍵によってデータを再暗号化して当該暗号化されたデータを上記遅延手段に供給し、上記遅延手段の遅延処理中に上記鍵サーバから新たな鍵が配布されなかったときは、上記遅延手段から出力される暗号化済みのデータを上記他のクライアント及び上記鍵サーバに送信する送信手段とを備えることを特徴とするネットワークシステム。

【請求項２】　上記各クライアントは、

上記送信手段からの暗号化済みのデータを上記他のクライアント及び上記鍵サーバに送信する第１の動作モードと、

上記送信対象となる暗号化済みのデータを上記他のクライアント及び上記鍵サーバに送信すると共に、当該暗号化済みのデータを最も遠いクライアントまで送信する時間に相当する時間記憶し、上記暗号化済みのデータの再送要求があったときは再送要求のあったクライアントに対して当該暗号化済みのデータを再送する第２の動作モードとを切換制御することを特徴とする請求項１記載のネットワークシステム。

【請求項３】　鍵サーバと複数のクライアントからなり、上記鍵サーバはクライアントが変更される毎に新たな共有鍵を変更後の各クライアントに送信し、各クライアントは配布された共有鍵を用いてデータを暗号化して暗号化済みのデータに対して鍵サーバと各クライアント間の因果順序を示す因果情報を付加して送受信を行うことで、共有鍵と暗号化済みのデータとを対応付けてデータの復号を行うネットワークシステムにおけるデータ送受信方法において、

上記各クライアントは、

上記鍵サーバに問い合わせをして当該鍵サーバから上記各クライアントに上記共有鍵が送信されるまでの保証されている遅延時間を算出し、

送信対象となる暗号化済みのデータに対して、上記算出された遅延時間に相当する時間の遅延処理を施し、

上記遅延処理中に上記鍵サーバから新たな鍵が配布されたときは、上記新たな鍵によってデータを再暗号化して当該暗号化されたデータを再び上記遅延処理の対象とし、上記遅延処理中に上記鍵サーバから新たな鍵が配布されなかったときは、上記遅延処理の施された暗号化済みのデータを上記他のクライアント及び上記鍵サーバに送信することを特徴とするデータ送受信方法。

【請求項４】　上記各クライアントは、

上記遅延処理中に上記鍵サーバから新たな鍵が配布されたときは、上記新たな鍵によってデータを再暗号化して当該暗号化されたデータを上記遅延処理対象とし、上記遅延処理中に上記鍵サーバから新たな鍵が配布されなかったときは、上記遅延処理の施された暗号化済みのデータを上記他のクライアント及び上記鍵サーバに送信する第１の動作モードと、

上記送信対象となる暗号化済みのデータを上記他のクライアント及び上記鍵サーバに送信すると共に、当該暗号化済みのデータを最も遠いクライアントまで送信する時間に相当する時間記憶し、上記暗号化済みのデータの再送要求があったときは再送要求のあったクライアントに対して当該暗号化済みのデータを再送する第２の動作モードとを切換制御することを特徴とする請求項３記載のデータ送受信方法。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、１つの共通鍵を用いてマルチキャスト通信を行うネットワークシステム及びデータ送受信方法に関する。

【０００２】

【従来の技術】インターネット等の広域分散ネットワークにおいてマルチキャスト技術は必要不可欠である。現在でもｍｂｏｎｅを始めとする仮想ネットワークでマルチキャスト通信は実現されているが、実際にエンドユーザに提供できる出力の品質はかなり低い。現在、ネットワークの高速化及び新技術の開発などに伴い、品質向上、さらに得ることのできるサービス品質の制御、Quality of Service（ＱｏＳ）技術の研究が盛んに行われている。そこで、今後マルチキャストアプリケーションはさらにリアルタイム性を向上することが求められている。

【０００３】鍵交換機構を必要とするマルチキャストアプリケーションは、通常、鍵を配布する鍵サーバとグループのメンバによって構成される。鍵サーバは、グループの一員であってもかまわない。通常、鍵はこのグループ単位で共有される。

【０００４】マルチキャストアプリケーションの場合、グループに参加しているメンバの構成は動的に変化する。すなわち、任意の時間において新たにメンバが参加したり、既存のメンバが脱退することが可能である。こ

のような場合、マルチキャスト通信によるデータが盗聴されることを防止するために新しい鍵を生成し、現存するメンバにその鍵を配布する必要がある。

【０００５】鍵交換機構を導入したマルチアプリケーションの場合、鍵の更新を行うための特別な処理が必要である。しかし、全てのメンバが同じ鍵を得るためには、一旦、アプリケーションの動作を止めることとなる同期をとり、鍵の一貫性を保証するための処理が必要であった。

【０００６】しかし、リアルタイムで動作するマルチキャストアプリケーションの場合、実行が中断されることは回避しなければならない。

【０００７】

【発明が解決しようとする課題】アプリケーションの実行を中断しない解決策としては、因果順序（Causal Ordering）を用いたものが提案されている。なお、以下の例では、鍵サーバとクライアントたるメンバＡ及びメンバＢによりネットワークが構成されているものとする。

【０００８】図９に示すように、鍵サーバは、メンバの構成が変わると、新しい鍵Ｋ’を古い鍵Ｋで暗号化して（{Ｋ’}Ｋ）、メンバＡ及びメンバＢに配布する。メンバＢは、新しい鍵Ｋ’を用いて送信すべきメッセージＭを暗号化して（{Ｍ}Ｋ’）、鍵サーバ及びメンバＡに送信している。ここで、メンバＡは、ネットワークの遅延によって、{Ｍ}Ｋ’を受け取ってから、新しい鍵Ｋ’を鍵サーバから受け取っている。メンバＡは、通常ではメンバＢからのメッセージＭを廃棄してしまうおそれがあるが、メンバＢにおいて付加された因果情報を参照することによって無事メッセージＭを新しい鍵で復号することができる。

【０００９】つぎに、図１０に示すように、メンバＢがメッセージＭについて鍵Ｋを用いて暗号化し（{Ｍ}Ｋ）、これを鍵サーバ及びメンバＡに送信した場合について説明する。このとき、鍵サーバは、メンバＢから{Ｍ}Ｋを受信してから、新たな鍵Ｋ’を古い鍵Ｋで暗号化して（{Ｋ’}Ｋ）、メンバＡ及びメンバＢに送信している。メンバＡは、鍵サーバから{Ｋ’}Ｋを受け取ってから、メンバＢからの{Ｍ}Ｋを受信している。メンバＡは、因果順序による順序づけを行うことによって、メンバＢから送られるメッセージ{Ｍ}Ｋを正しく復号することができる。すなわち、アプリケーションを中断し、最新鍵の同期をとる必要はない。

【００１０】しかし、図１１の場合は、問題が発生する。すなわち、鍵サーバが{Ｋ’}Ｋをメンバ Ａ及びメンバＢに配布し、その直後にメンバＢが{Ｍ}Ｋを鍵サーバ及びメンバＡに送信し、メンバＡが{Ｋ’}Ｋの後に{Ｍ}Ｋを受信した場合である。

【００１１】このとき、鍵サーバは新しい鍵Ｋ’を送信し、さらにメンバＢは古い鍵Ｋで暗号化したメッセージＭを送信する。しかし、この２つの転送に対しては因果

情報を付加することができない。そこで、メンバＡは、復号できない{Ｍ}に対する再送要求をメンバＢに発行し、新たな鍵Ｋ’で暗号化されたメッセージＭを送信するように要求する。

【００１２】このような再送要求機構を用いることによって、一時的には上述した問題を解決することができる。しかし、実際には、メンバＢは、再送要求に備えてメッセージＭを保持しておく必要がある。メンバＡがどの程度再送要求を発生するかを予測することはできないので、メンバＢが保持しておくメッセージＭの数を決定することも困難である。

【００１３】ネットワーク遅延を決定することできない場合は、理論的には無限のメッセージを保持しなければならない。この場合、メンバＡは、{Ｍ}Ｋの再送要求を行ったことによって、再度メッセージＭを受け取るまでの予測不可能な遅延が発生する。これにより、リアルタイム性を維持することができなくなり、また、連続的にアプリケーションにデータを供給することができなくなる。

【００１４】本発明は、このような実情に鑑みて提案されたものであり、鍵の配布や鍵を用いたデータの送受信の順番に関係なく、リアルタイム性を維持してマルチキャスト通信を行うことができるネットワークシステム及びデータ送受信方法を提供することを目的とする。

【００１５】

【課題を解決するための手段】上述の課題を解決するために、本発明に係るネットワークシステムは、鍵サーバと複数のクライアントからなり、鍵サーバはクライアントが変更される毎に新たな共有鍵を変更後の各クライアントに送信し、各クライアントは配布された共有鍵を用いてデータを暗号化して暗号化済みのデータに対して鍵サーバと各クライアント間の因果順序を示す因果情報を付加して送受信を行うことで、共有鍵と暗号化済みのデータとを対応付けてデータの復号を行うネットワークシステムにおいて、各クライアントは、鍵サーバに問い合わせをして当該鍵サーバから各クライアントに共有鍵が送信されるまでの保証されている遅延時間を算出する算出手段と、送信対象となる暗号化済みのデータに対して、算出手段により算出された遅延時間に相当する時間の遅延処理を施す遅延手段と、遅延手段の遅延処理中に鍵サーバから新たな鍵が配布されたときは、新たな鍵によってデータを再暗号化して当該暗号化されたデータを遅延手段に供給し、遅延手段の遅延処理中に鍵サーバから新たな鍵が配布されなかったときは、遅延手段から出力される暗号化済みのデータを他のクライアント及び鍵サーバに送信する送信手段とを備えることを特徴とする。

【００１６】本発明に係るデータ送受信方法は、鍵サーバと複数のクライアントからなり、上記鍵サーバはクライアントが変更される毎に新たな共有鍵を変更後の各ク

ライアントに送信し、各クライアントは配布された共有鍵を用いてデータを暗号化して暗号化済みのデータに対して鍵サーバと各クライアント間の因果順序を示す因果情報を付加して送受信を行うことで、共有鍵と暗号化済みのデータとを対応付けてデータの復号を行うネットワークシステムにおけるデータ送受信方法において、上記各クライアントは、上記鍵サーバに問い合わせをして当該鍵サーバから上記各クライアントに上記共有鍵が送信されるまでの保証されている遅延時間を算出し、送信対象となる暗号化済みのデータに対して、上記算出された遅延時間に相当する時間の遅延処理を施し、上記遅延処理中に上記鍵サーバから新たな鍵が配布されたときは、上記新たな鍵によってデータを再暗号化して当該暗号化されたデータを再び上記遅延処理の対象とし、上記遅延処理中に上記鍵サーバから新たな鍵が配布されなかったときは、上記遅延処理の施された暗号化済みのデータを上記他のクライアント及び上記鍵サーバに送信することを特徴とする。

【００１７】

【発明の実施の形態】以下、本発明の実施の形態について、図面を参照しながら説明する。

【００１８】本発明は、例えば図１に示す構成のネットワークシステム１に適用することができる。

【００１９】上記ネットワークシステム１は、ネットワーク間で１つの鍵を共有するマルチキャスト通信を行うものであり、クライアントたるメンバ２（２Ａ，２Ｂ，２Ｃ，・・・）と暗号化／復号のための鍵を配布する鍵サーバ３とがネットワーク４を介して接続されている。

【００２０】マルチキャスト通信は、第三者の盗聴を防止するために、新たなメンバが参加したりメンバが脱退したときには、共有している鍵（以下、「共有鍵」という。）の更新を行う。このような共有鍵の更新は、鍵サーバ３が新たな共有鍵を生成して各メンバに配布することによって行われる。なお、鍵サーバ３によって配布される共有鍵が全てのメンバ２に送信されることを保証するために、鍵配布のためのマルチキャスト通信は信頼性のあるものを用いる。さらに、鍵サーバ３が各メンバ２間において新しい共有鍵を配布するために必要な時間ｔは、予約資源プロトコル等によって保証されているものとする。

【００２１】メンバ２は、図２に示すように、共有鍵を用いた暗号化／復号等の所定のデータ処理等を行うアプリケーション２１と、因果順序マルチキャスト機構２２とを備える。因果順序マルチキャスト機構２２は、暗号化済みのデータを送信する送信モジュール２３と、暗号化されたデータや共有鍵を受信する受信モジュール２４とを備える。

【００２２】送信モジュール２３は、暗号化されたデータに遅延を与える遅延キュー２５と、遅延キュー２５からのデータに因果順序を示すデータであるベクタタイムスタンプを挿入するベクタタイムスタンプ機構２６とを有する。

【００２３】遅延キュー２５は、アプリケーション２１からのデータに対して、常に一定時間の遅延を与えてから出力するように制御されている。ここで、一定時間とは、鍵サーバ３が各メンバ２に共有鍵を配布するまでの時間ｔをいう。

【００２４】ベクタタイムスタンプ機構２６は、ベクタタイムスタンプを生成し、これを遅延キュー２５からのデータに挿入して、他のメンバ２及び鍵サーバ３に送信する。

【００２５】以下、プロセス（メンバ）ｐｉのベクタタイムスタンプをＶＴ（ｐｉ）と表記し、プロセスｐｉのベクトル要素をＶＴ（ｐｉ）[i]で表す。また、送信されるデータに挿入するベクタタイムスタンプをＶＴ（ｍ）と表記する。なお、ここで用いるベクタタイムスタンプは[1]に基づくものとする。

【００２６】ベクタタイムスタンプの更新アルゴリズム以下に示す。

１．プロセス起動時にＶＴ（ｐｉ）の全要素を０に初期化する。

【００２７】２．送信時にＶＴ（ｐｉ）[i]を１増加する。

【００２８】３．送信データにＶＴ（ｐｉ）を挿入する（以下、これを「ＶＴ（ｍ）」とする）。

【００２９】４．ｐｉからのデータをｐｊが受信した場合には（１）式に従って、ＶＴ（ｐｊ）を　更新する。

【００３０】

【数１】

$$VT(p_j)[k] = \max(VT(p_j)[k], VT(m)[k]) \quad (1)$$

ただし，n をメンバ数とすると，$\{\forall k : k \in n\}$ である．

【００３１】一方、受信モジュール２４は、受信したデータに挿入されているベクタタイムスタンプを検査するベクタタイムスタンプ機構２７と、アプリケーション２１へのデータの配送を一時抑止する抑止キュー２８と、抑止キュー２８からのデータを鍵サーバアプリケーション２１に配送する配送キュー２９とを備える。

【００３２】ベクタタイムスタンプ機構２７は、データを受信したときに、当該データに挿入されているベクタタイムスタンプが（２）式の条件を満たしているかを判定する。

【００３３】

【数２】

$$\begin{cases} VT(m)[k] = VT(p_j)[k] + 1 & k = i \\ VT(m)[k] \leq VT(p_j)[k] & k \neq i \end{cases} \quad (2)$$

ただし，nをメンバ数とすると，$\{\forall k : k \in n\}$である．

【００３４】ベクタタイムスタンプ機構２７は、（２）式の条件を満たしていない場合には、受信したデータを抑止キュー２８に供給する。また、ベクタタイムスタンプ機構２７は、（２）式の条件を満たしている場合は、配送キュー２９を介してアプリケーション２１に供給し、ベクタタイムスタンプを（１）式に従って更新する。

【００３５】つぎに、このようなベクタタイムスタンプの更新アルゴリズムについて、簡単な具体例を挙げながら説明する。図３に示すように、メンバがＡ，Ｂ，Ｃからなるグループがあるものとする。Ａ，Ｂ，Ｃは、それぞれ起動時には、ベクタタイムスタンプの全要素が０である（０，０，０）を持つ。

【００３６】Ａは、Ｂ及びＣに対してデータを送信するとき、（１，０，０）をＶＴ（ｍ）として上記データに挿入し、当該データをＢ及びＣに送信する。このとき、自己のＶＴ（Ａ）も（１，０，０）に更新する。

【００３７】Ｃは、ベクタタイムスタンプ機構２７でＡからのデータを受信すると、自己のＶＴ（Ｃ）とＶＴ（ｍ）とを比較する。このとき（２）式を満足するので、ベクタタイムスタンプ機構２７は、データを配送キュー２９に供給し、ＶＴ（Ｃ）を（１，０，０）に更新する。その後、Ｃがデータを送信する場合には、ＶＴ（ｍ）を（１，０，１）としてデータに挿入し、Ａ及びＢに送信する。

【００３８】ここで、Ｂは、Ｃからのデータを受信した後、Ａからのデータを受信する。Ｂは、Ｃからのデータをベクタタイムスタンプ機構２７で受信し、そのベクタタイムスタンプが（２）式を満たしているか判定する。ここでは、（２）式を満たさないので、ベクタタイムスタンプ機構２７は、上記データを抑止キュー２８に供給する。

【００３９】Ｂは、Ａからのデータを受信する。Ｂのベクタタイムスタンプ機構２７は、ＶＴ（ｍ）が（１，０，０）であるＡからのデータを受信すると（２）式を満足するので、Ａからのデータを配送キュー２９に供給する。このとき、ＶＴ（Ｂ）は（１，０，０）に更新され、抑止キュー２８に抑止されていたデータは、（２）式を満足するようになる。そして、抑止キュー２８は、抑止していたＣからのデータを配送キュー２９を介してアプリケーション２１に供給する。

【００４０】これにより、Ｂは、ＡからのデータがＣからのデータに遅れて到着した場合であっても、ベクタタイムスタンプを用いて因果順序を満足させて、Ａからのデータ，Ｃからのデータの順にアプリケーション処理を行うことができる。

【００４１】なお、ベクタタイムスタンプの要素数は、グループのメンバシップが変わる度に変化する。具体的には、１のメンバが新たに参加するとベクタタイムスタンプの要素は１つ大きくなり、１のメンバが脱退するとその要素は１つ小さくなる。

【００４２】さらに、メンバ２のシステムとして、リライアブルマルチキャスト・資源予約プロトコル機構５１と、ＩＰマルチキャスト機構５２とを備えている。

【００４３】リライアブルマルチキャスト・資源予約プロトコル機構５１は、マルチキャスト通信に信頼性を保証するものである。ここにいう信頼性とは、第１に、共有鍵が全てのメンバ２に送信されること、第２に、鍵サーバ３が各メンバ２間において新しい共有鍵を配布するために必要な時間が予約資源等によって保証されていることである。

【００４４】ＩＰマルチキャスト機構５２は、マルチキャスト通信用のインターネットプロトコルであり、上記リライアブルマルチキャスト・資源予約プロトコル機構５１からのデータを他のメンバ２や鍵サーバ３に転送したり、また、受信したデータや共有鍵をリライアブルマルチキャスト・資源予約プロトコル機構５１に供給する。

【００４５】鍵サーバ３は、メンバ２とほぼ同様の構成となっている。すなわち、鍵サーバ３は、鍵サーバアプリケーション３１と、因果順序マルチキャスト機構３２とを備える。

【００４６】鍵サーバアプリケーション３１は、グループに参加しようとするユーザから参加メッセージを受信したりメンバが脱退したときのメッセージを受信したりすると、新たな共有鍵を生成し、この共有鍵を送信モジュール３３に供給する。

【００４７】送信モジュール３３は、遅延キュー３５と、ベクタタイムスタンプ機構３６とを備え、送信モジュール２３と同様の構成である。

【００４８】遅延キュー３５は、システムの実装上設けてあるだけであり、共有鍵の配布時に当該共有鍵に遅延を与えるものではない。すなわち、鍵サーバアプリケーション３１からの共有鍵は、すぐにベクタタイムスタンプ機構３６に供給される。

【００４９】ベクタタイムスタンプ機構３６は、鍵サーバアプリケーション３１からの共有鍵にベクタタイムスタンプを挿入して、各メンバ２に送信する。

【００５０】受信モジュール３４は、暗号化されたデータに挿入されたベクタタイムスタンプを検査するベクタタイムスタンプ機構３７と、上記データを上述した（２）式の条件を満たすまで抑止する抑止キュー３８と、抑止キュー３８から供給されるデータを鍵サーバアプリケーション３１に配送する配送キュー３９とを備

え、ベクタタイムスタンプ機構２７と同様の構成となっている。

【００５１】また、鍵サーバ３のシステムは、リライアブルマルチキャスト・資源予約プロトコル機構５１及びＩＰマルチキャスト機構５２を備えている。

【００５２】すなわち、鍵サーバ３も、ネットワークシステム１を構成するメンバの一員であり、他のメンバ２と同様に、共有鍵にベクタタイムスタンプを挿入して各メンバに配布したり、共有鍵によって暗号化されたデータを受信している。

【００５３】このようなネットワークシステム１は、第三者に盗聴されることなく安全にリアルタイムマルチキャスト通信を行うことができるように、このグループを構成するメンバの変化に応じた処理を行っている。

【００５４】図４に示すように、ユーザがメンバとして上記ネットワークシステム１のグループに参加すると（ステップＳＴ１）、上記メンバ２は、鍵サーバ３とやり取りを行う（ステップＳＴ２）。

【００５５】例えば、メンバとして参加するときは、鍵サーバ３に対して参加メッセージを送信する。鍵サーバ３は、参加メッセージを受信すると、他のメンバに対して新たな共有鍵を送信し、上記参加メッセージを送信したものに対して鍵の配布を行う。鍵が配布されることによって、メンバとして登録されたことになる。

【００５６】新たに参加したメンバ２は、鍵サーバ３に対して問い合わせを行う。具体的には、メンバ２の因果順序マルチキャスト機構２２が、鍵サーバ３の因果順序マルチキャスト機構３２に対して、鍵サーバ３から各メンバ２において新しい共有鍵を配布するための必要な時間の問い合わせを行い、共有鍵を配布するための必要な時間ｔを算出して、時間ｔを決定する（ステップＳＴ３）。なお、因果順序マルチキャスト機構２２は、上記時間ｔを算出することができるのであれば、例えば鍵サーバ３のシステムであるリライアブルマルチキャスト・資源予約プロトコル機構５１やその他の機構に直接問い合わせても良い。

【００５７】その後、メンバ２がグループから脱退するときは図５に示すステップＳＴ１１の処理に移行し、メンバ２がデータを受信するときは図６に示すステップＳＴ２１に移行し、メンバ２がデータを送信するときは図７に示すステップＳＴ３１に処理に移行する。

【００５８】メンバ２は、図５に示すように、当該グループから脱退するときは（ステップＳＴ１１）、鍵サーバ３に対して脱退メッセージを通知する（ステップＳＴ１２）。鍵サーバ３は、上記脱退メッセージを受信すると、脱退するメンバ２を除いた各メンバ２に対して新たな鍵を配布する。これにより、メンバ２は、完全に脱退したことになる。

【００５９】一方、メンバ２は、データを受信するときは、図６に示すステップＳＴ２１以下の処理を実行する。

【００６０】ステップＳＴ２１において、メンバ２のベクタタイムスタンプ機構２７は、他のメンバ２からのデータを受信すると、ステップＳＴ２２に進む。

【００６１】ステップＳＴ２２において、ベクタタイムスタンプ機構２７は、受信したデータに挿入されているベクタタイムスタンプの内容を検査し、因果関係があるかを判定する。ベクタタイムスタンプ機構２７は、因果関係がない、すなわち受信したデータは因果順序に従っていないと判定したときはステップＳＴ２３に進み、因果関係がある、すなわち受信したデータは因果関係に従っていると判定したときはステップＳＴ２４に進む。

【００６２】ステップＳＴ２３において、ベクタタイムスタンプ機構２７は、受信したデータを抑止キュー２８に供給して、ステップＳＴ２１に戻る。すなわち、次々に受信するデータが因果順序に従っていないときは、ステップＳＴ２１からステップＳＴ２３の処理が繰り返しなされ、次々にデータが抑止キュー２８に抑止されることになる。

【００６３】ステップＳＴ２４において、メンバ２のシステムのＶＴ（ｐ）を更新して、ステップＳＴ２５に進む。

【００６４】ステップＳＴ２５において、ベクタタイムスタンプ機構２７は、受信したデータを配送キュー２９を介してアプリケーション２１に供給し、ステップＳＴ２６に進む。これにより、アプリケーション２１は、自身が保有する共有鍵を用いてデータの復号を行い、さらに、所定の処理を実行する。

【００６５】ステップＳＴ２６において、抑止キュー２８に抑止されているデータがあるかを判定する。抑止キュー２８にデータがあるときはステップＳＴ２７に進み、抑止キュー２８にデータがないときは、メンバ２におけるデータ受信処理を終了する。

【００６６】ステップＳＴ２７において、抑止キュー２８に抑止されているデータの因果関係を確認し、因果関係がある、すなわち抑止キュー２８に抑止されているデータが因果順序に従っていると判定したときはステップＳＴ２８に進む。また、因果関係がない、すなわち上記データは因果順序に従っていないと判定したときはステップＳＴ３０に進む。

【００６７】ステップＳＴ２８において、メンバ２のシステムのＶＴ（ｐ）を更新して、ステップＳＴ２８に進む。

【００６８】ステップＳＴ２９において、抑止キュー２８は、抑止していたデータを配送キュー２９を介してアプリケーション２１に供給し、ステップＳＴ２６に戻る。

【００６９】一方、ステップＳＴ２７で因果関係がないと判定したときのステップＳＴ３０において、抑止キュー２８に抑止されている次の要素、すなわち次のデータ

を処理の対象として、ステップＳＴ２６に戻る。これにより、抑止キュー２８に抑止されているデータを次々に処理する。

【００７０】以上のように、メンバ２は、他のメンバ２から送信されるデータについて、ベクタタイムスタンプを検査して因果順序を判断することによって、正確な順序でデータを受信することができる。なお、メンバ２が鍵サーバ３から共有鍵を受信した場合も同様である。さらに、鍵サーバ３がメンバ２からデータを受信する場合も同様の処理を実行する。

【００７１】また、メンバ２は、データを他のメンバ２に送信するときは、図７に示すステップＳＴ３１以下の処理を実行する。

【００７２】ステップＳＴ３１において、アプリケーション２１は送信すべきデータの暗号化処理を行い、受信モジュール２４はこれを受け取って、ステップＳＴ３２に進む。

【００７３】ステップＳＴ３２において、受信モジュール２４は暗号化済みのデータを遅延キュー２５に供給して、ステップＳＴ３３に進む。

【００７４】ステップＳＴ３３において、遅延キュー２５にデータが供給されてから出力するまでの時間ｔの間に共有鍵が配布されたかを判定し、共有鍵が配布されたと判定したときはステップＳＴ３４に進み、共有鍵が配布されていないと判定したときはステップＳＴ３６に進む。

【００７５】ステップＳＴ３４において、メンバ２は、保有している共有鍵を新たに配布された共有鍵に更新して、ステップＳＴ３５に進む。ここで、配布された新しい共有鍵は、古い共有鍵によって暗号化されている。したがって、アプリケーション２１が古い鍵を用いて新しい鍵を復号することで、新たな鍵の更新を行う。

【００７６】ステップＳＴ３５において、遅延キュー２５に記憶されているデータの送信を中止すると共に、アプリケーション２１は新たな共有鍵を用いて送信の対象となっていたデータを再び暗号化し直して、ステップＳＴ３１に戻る。

【００７７】一方、ステップＳＴ３３で時間ｔの間に共有鍵が配布されていないと判定したときのステップＳＴ３６において、遅延キュー２５に記憶されているデータを他のメンバ２や鍵サーバ３に送信し、送信処理を終了する。

【００７８】このようなステップＳＴ３１からステップＳＴ３６の処理を行うことによって、図１１で説明した問題を解決することができる。

【００７９】例えば図８に示すように、鍵サーバ（Key Server），メンバＡ（Member A），メンバＢ（Member B）からなるグループがあるものとする。鍵サーバは、新しい共有鍵K_{new}を生成すると、古い共有鍵K_{old}を用いて当該新しい共有鍵K_{new}を暗号化し（$\{K_{new}\}K_{old}$）、これをメンバＡ及びメンバＢに配布する。

【００８０】一方、メンバＢのアプリケーション２１は、古い共有鍵K_{old}を用いてメッセージＭを暗号化し、これを遅延キュー２５に供給する。遅延キュー２５は、時刻ｔ１において、アプリケーション２１から暗号化されたメッセージＭを受け取って保持し、時間ｔのバッファリングを開始する。

【００８１】時刻ｔ２において、メンバＢは、鍵サーバが配布した$\{K_{new}\}K_{old}$を受信すると、遅延キュー２５に保持されていたデータの送信を中止する。このとき、アプリケーション２１は、$\{K_{new}\}K_{old}$を復号して、新しい共有鍵K_{new}を用いてメッセージＭを再暗号化する（$\{M\}K_{new}$）。遅延キュー２５は、$\{M\}K_{new}$を保持すると、再び時間ｔのバッファリングを開始する。

【００８２】時刻ｔ３において、メンバＢは、時間ｔのバッファリングが終了すると、$\{M\}K_{new}$を鍵サーバ及びメンバＡに送信する。

【００８３】これにより、メンバＡは、鍵サーバからの新たな鍵$\{K_{new}\}K_{old}$を受け取ってから、メンバＢからの新たな共有鍵によって暗号化されたデータ$\{M\}K_{new}$を受信するので、再送要求をすることなくメンバＢからのデータを復号することができる。

【００８４】すなわち、上記ネットワークシステム１は、このような場合にメンバＡがメンバＢに対して再送要求を行うことがなくなるので、グループ内のデータの秘密性を保持しつつ、常にリアルタイムでマルチキャスト通信を行うことができる。

【００８５】また、上記ネットワークシステム１は、さらにデータの再送要求機構を備えるようにしてもよい。通常、各メンバ２は再送要求に備えて送信済みのデータを一時保存しておく必要があり、従来は、どのくらい保存しておけば良いか不明であり、場合によっては無限に保存しておく必要があった。

【００８６】しかし、各メンバ２は、最も遠い受信者たるメンバ２又は鍵サーバ３までの転送時間Ｔを算出して転送時間Ｔの間だけデータを保存しておき、再送要求があったときには再送要求先に上記データを再送すればよい。これにより、各メンバ２が無用に多くのデータを保存する負担を減らすことができる。また、各メンバ２は、再送要求の代わりに、復号できなかったデータの有無をアプリケーションに通知するようにしてもよい。

【００８７】さらに、上記ネットワークシステム１は、状況に応じて、上述した送信データのバッファリング機構と、再送要求機構とのいずれかを選択して、処理を実行するようにしてもよい。これにより、各メンバ２の使用状況やシステムの処理能力に応じたマルチキャスト通信を行うことができる。

【００８８】なお、本発明は、リアルタイムで多数の人が対戦することができるオンラインゲーム、チャット等にも適用することができる。また、高性能な動画を転送

するビデオ会議システム、マルチユーザが参加する仮想空間分散ゲーム等、リアルタイムマルチキャスト通信が使用される場合に適用することができるのは勿論である。

【００８９】

【発明の効果】以上詳細に説明したように、本発明に係るネットワークシステム及びデータ送受信方法によれば、各クライアントは、送信対象となる暗号化済みのデータに対して、保証遅延時間に相当する時間の遅延処理を施し、上記遅延処理中に鍵サーバから新たな鍵が配布されたときは、新たな鍵によってデータを再暗号化して当該暗号化されたデータを再び遅延処理の対象とし、上記遅延処理中に鍵サーバから新たな鍵が配布されなかったときは、遅延処理の施された暗号化済みのデータを他のクライアント及び鍵サーバに対して送信することにより、共有鍵の配布や共有鍵を用いたデータの送受信の順番に関係なく、データの秘密性を保持しつつ、リアルタイム性を維持してマルチキャスト通信を行うことができる。

【図面の簡単な説明】

【図１】本発明を適用したネットワークシステムの概略的な構成を示す図である。

【図２】上記ネットワークシステムの具体的なシステム構成を示すブロック図である。

【図３】因果順序による順序付けを説明するための一例を示した図である。

【図４】上記ネットワークシステムによって構成されるグループにあるユーザがメンバとして参加するときの動作を示すフローチャートである。

【図５】メンバが上記グループから脱退するときの動作を示すフローチャートである。

【図６】メンバが他のメンバから暗号化されたデータを受信するときの動作を示すフローチャートである。

【図７】メンバが他のメンバに対してデータを暗号化して送信するときの動作を示すフローチャートである。

【図８】鍵サーバ、メンバＡ、メンバＢからなるグループにおいて、メンバＢがバッファリングしてデータを送信することを説明するための図である。

【図９】鍵サーバ、メンバＡ、メンバＢからなるグループにおいて、メンバＡが、因果順序による順序付けをしてデータを受信することを説明する図である。

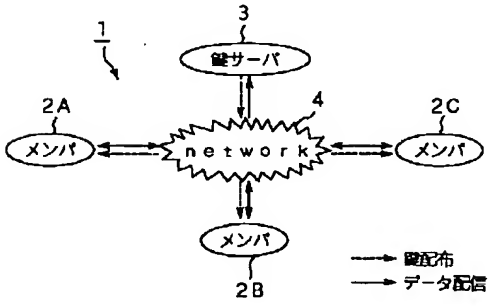
【図１０】上記グループにおいて、メンバＡが、因果順序による順序付けをしてデータを受信することを説明する図である。

【図１１】上記グループにおいて、メンバＡが、データを受信する際に、因果順序による順序付けを行うことができない場合を説明する図である。

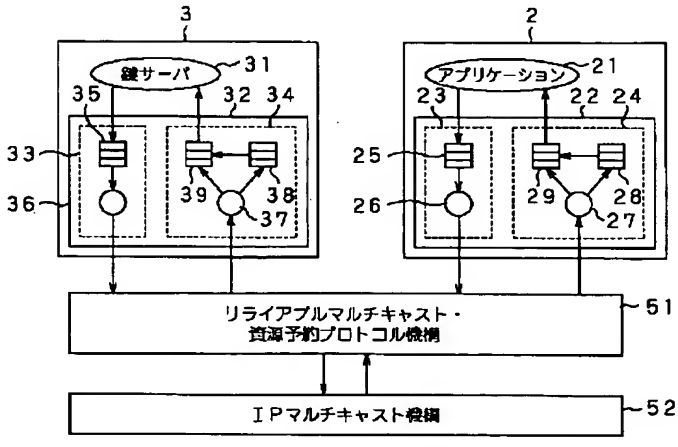
【符号の説明】

１　ネットワークシステム、２　メンバ、３　鍵サーバ、２１　アプリケーション、２２，３２　因果順序マルチキャスト機構、２３，３３　送信モジュール、２４，３４　受信モジュール、２５，３５　遅延キュー、２６，３６　ベクタタイムスタンプ機構、２７，３７　ベクタタイムスタンプ機構、２８，３８　抑止キュー、２９，３９　配送キュー、５１　リライアブルマルチキャスト・資源予約プロトコル機構、５２　ＩＰマルチキャスト機構

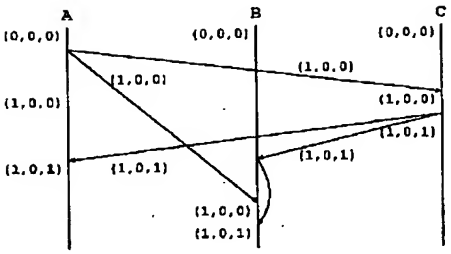
【図１】



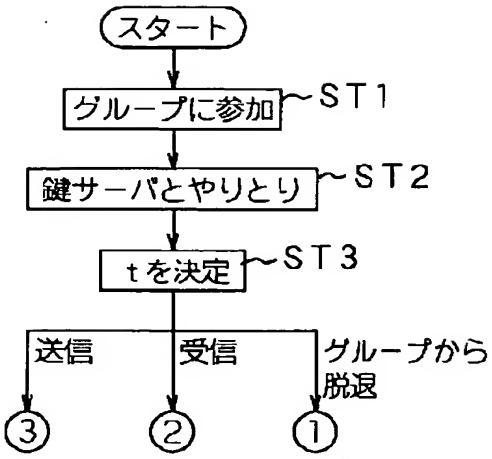
【図２】



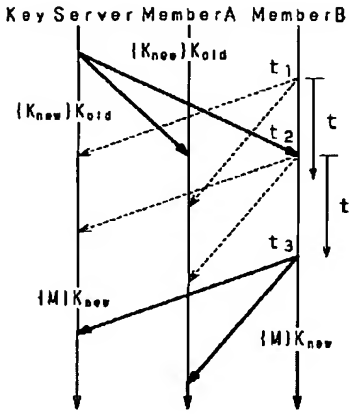
【図３】



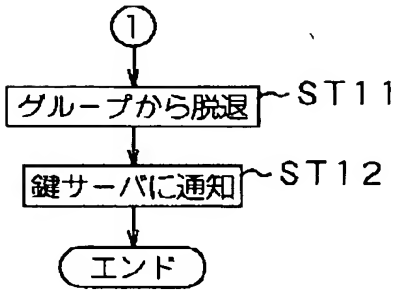
【図４】



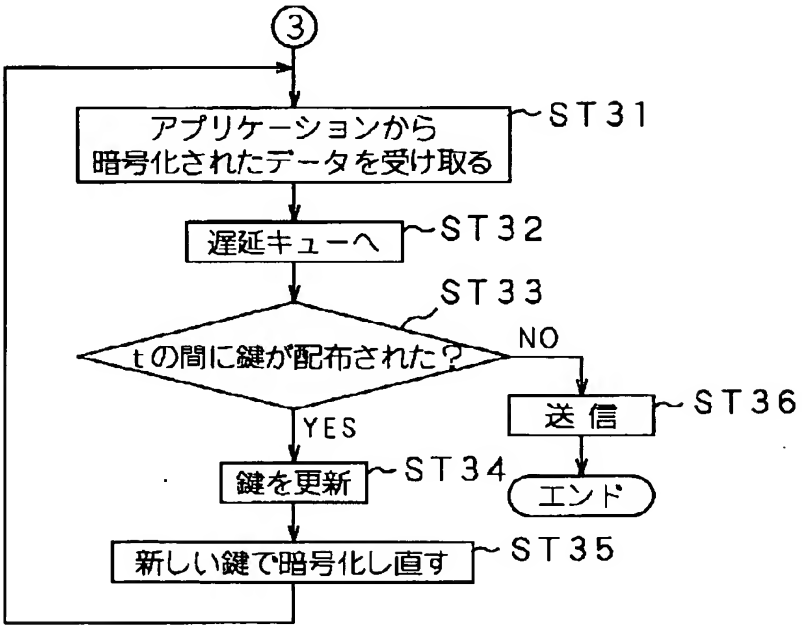
【図８】



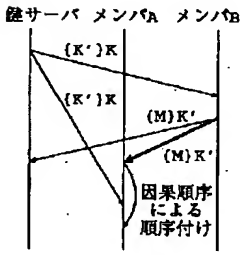
【図５】



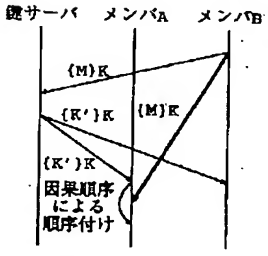
【図７】



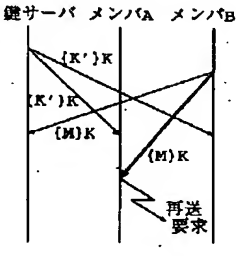
【図９】



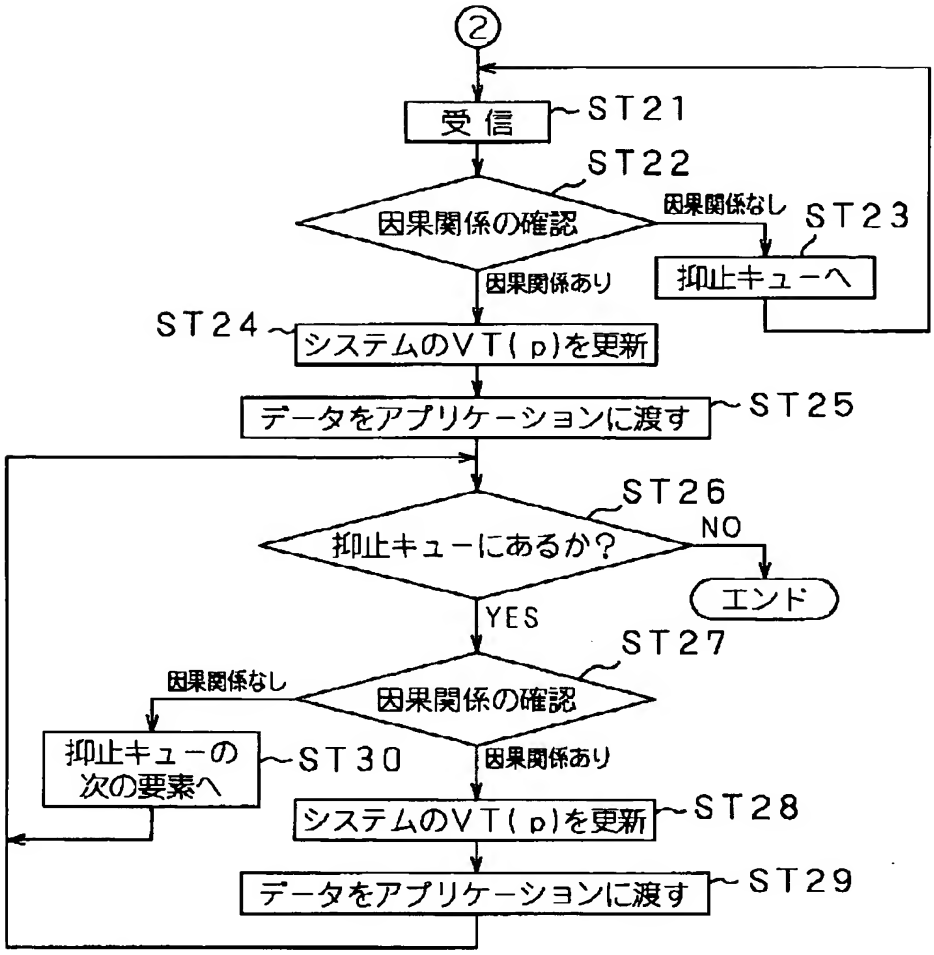
【図１０】



【図１１】



【図６】



フロントページの続き

Ｆターム(参考) 5J104 AA01 AA16 AA34 EA01 EA16
MA05 NA02 PA07
5K030 GA15 HA08 HB16 HB18 HC01
JT02 JT06 LA07 LD06 LD19
9A001 CC03 EE03 JJ13 JJ18 KK60
LL03